Citation for published version:
Carley, MJ 2010, 'Moving least squares via orthogonal polynomials', *SIAM Journal on Scientific Computing*, vol. 32, no. 3, pp. 1310-1322. https://doi.org/10.1137/09076711X

DOI:
10.1137/09076711X

Publication date:
2010

Link to publication

©SIAM

University of Bath

Alternative formats
If you require this document in an alternative format, please contact:
openaccess@bath.ac.uk

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Download date: 05. May. 2023

SIAM J. SCI. COMPUT.
Vol. 32, No. 3, pp. 1310–1322
© 2010 Society for Industrial and Applied Mathematics

MOVING LEAST SQUARES VIA ORTHOGONAL POLYNOMIALS*

MICHAEL CARLEY†

Abstract. A method for moving least squares interpolation and differentiation is presented in the framework of orthogonal polynomials on discrete points. This yields a robust and efficient method which can avoid singularities and breakdowns in the moving least squares method caused by particular configurations of nodes in the system. The method is tested by applying it to the estimation of first and second derivatives of test functions on random point distributions in two and three dimensions and by examining in detail the evaluation of second derivatives on one selected configuration. The accuracy and convergence of the method are examined with respect to length scale (point separation) and the number of points used. The method is found to be robust, accurate, and convergent.

Key words. moving least squares, interpolation, numerical differentiation, orthogonal polynomials

AMS subject classifications. 65D25, 65D05, 42C05

DOI. 10.1137/09076711X

1. Introduction.

The moving least squares method [3, Chapter 7] is a technique for interpolation [6] and differentiation [1, 2, 7, 8, 9, 10, 13] on scattered data. The purpose of this paper is to examine the moving least squares problem in the framework of orthogonal polynomials, as applied to the estimation of derivatives.

In applications of the type considered here, the data supplied are the positions of N points $\mathbf{x}_i$, $i = 1, \ldots, N$, and corresponding values f_i. At one of these points the derivative is to be estimated. This is done using an interpolating polynomial $P(\mathbf{x})$ which minimizes the error

$$E = \sum_{i=1}^{N} w_i \left(P(\mathbf{x}_i) - f_i \right)^2 , \tag{1.1}$$

where w_i is a strictly positive weight. The polynomial $P(\mathbf{x})$ can be computed by a direct solution of a least squares problem and then used to interpolate $f(\mathbf{x})$ or to estimate its derivatives.

There are a number of applications where moving least squares is used to estimate derivatives of a function specified at discrete points. One is the estimation of gradients of vorticity in Lagrangian vortex methods [7, 8], where the gradients are estimated in two or three dimensions by fitting a second order polynomial to the components of vorticity and differentiating the polynomial. It was noted that "when computational points become very isolated, due to inadequate spatial resolution, the condition number of the matrix [used in fitting the polynomial] becomes very large" [7]. The solution proposed for this ill-conditioning was to add additional points to the fit. It appears that this problem may have been caused by another effect which has been

*Received by the editors August 4, 2009; accepted for publication (in revised form) March 22, 2010; published electronically May 5, 2010. Part of this work was carried out at the Institut für Strömungsmechanik und Hydraulische Strömungsmaschine of the University of Stuttgart, Germany, as part of the European Community–funded project HPC-Europa, contract 506079.
http://www.siam.org/journals/sisc/32-3/76711.html
†Department of Mechanical Engineering, University of Bath, Bath BA2 7AY, United Kingdom (m.j.carley@bath.ac.uk).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

noted by authors who use moving least squares to solve partial differential equations on irregular meshes or using mesh-free techniques.

In the work of Schönauer and Adolph [9,10], a finite difference stencil is developed using polynomials which interpolate data on points of an unstructured mesh. The points used in the polynomials are selected by choosing more points than there are coefficients in the fitting polynomial because "in m nodes usually there is not sufficient information for the m coefficients" [9] or, restated, "there are linear dependencies on straight lines" [10]. The number of extra points used in fitting the polynomial was determined through experience and testing. This raises the issue of the arrangement of the points used in deriving a polynomial fit.

The issue has been addressed recently by Chenoweth, Soria, and Ooi [2] who considered the problem of how to find a least squares fit on points of an unstructured mesh in order to generate a stencil, while avoiding singularities caused by particular point configurations, a general form of the problem of "linear dependencies" [10]. They state the conditions under which such singularities will arise and state a criterion determining when it will not be possible to make a least squares fit of a given order on a given set of points in two dimensions. This will happen when selected points are spanned by the same polynomial, for example, when fitting a second order polynomial to points which lie on an ellipse in two dimensions. They also give an algorithm for a moving least squares fit which determines when more points must be added in order to avoid singularities, and which additional points will be useful.

Another recent paper employing moving least squares methods for three-dimensional meshless methods [13] proposes an approach which may help avoid the problem of singularities. The method is to derive a set of basis functions which are orthogonal with respect to an inner product defined on the set of points. The use of orthogonal functions has the advantage of improving the condition number of the system to be solved to form the least squares fit and, in this case, allows a smaller number of basis functions to be used. The authors do not, however, discuss the problem of singular point configurations other than stating the number of points included in the fit must be large enough to make the system matrix regular, which corresponds to the avoidance of singular or ill-conditioned arrangements of nodes.

Strangely, there does not yet appear to be a published moving least squares method which explicitly frames the problem in terms of orthogonal polynomials. The aim of this paper is to present a method using results from the theory of orthogonal polynomials in multiple variables [11] to restate the problem in a manner which detects singular point configurations and generates a set of orthogonal polynomials which are unique for the points considered. The polynomials derived can then be used directly in computing a fit to the function on the specified data points. The method is quite general and does not require a knowledge of which configurations give rise to singularities. In three or more dimensions these configurations are not easily visualized, and, furthermore, a singular value decomposition becomes increasingly expensive.

2. Discrete orthogonal polynomials for scattered data. The theory of classical orthogonal polynomials of several variables is well-developed [11] but that of polynomials orthogonal on discrete points is not as advanced. A recent paper [12], however, establishes basic properties of discrete orthogonal polynomials and gives algorithms for their derivation. In particular, it establishes the theoretical foundations which allow us to say, given a set of points, whether orthogonal polynomials of a given order exist on these points and, if they do, what those polynomials are. In this section, we will summarize the mathematical tools required to derive and apply polynomials

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

orthogonal on discrete points. We use the standard notation in which a polynomial of several variables is defined as a weighted sum of monomials:

$$P(\mathbf{x}) = \sum_{i=1}^{n} A_i \mathbf{x}^{\boldsymbol{\alpha}_i}, \tag{2.1}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_d)$, $\mathbf{x} \in \mathbb{R}^d$, $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_d)$, $\boldsymbol{\alpha} \in \mathbb{N}_0^d$, and the monomial terms $\mathbf{x}^{\boldsymbol{\alpha}} = \prod_{j=1}^{d} x_j^{\alpha_j}$. The degree of $P(\mathbf{x})$ is $\max |\boldsymbol{\alpha}_i|$, where $|\boldsymbol{\alpha}| = \sum_{j=1}^{d} \alpha_j$.

2.1. Generation of orthogonal polynomials. The first basic tool required is a scheme to generate a set of polynomials which are orthogonal on a given set of points with respect to some weight function. This can be done using standard matrix operations [4, 5] using the procedure given by Xu [12]. First, we define the inner product

$$\langle f(\mathbf{x}), g(\mathbf{x}) \rangle = \sum_{i=1}^{N} f(\mathbf{x}_i) g(\mathbf{x}_i) w_i,$$

where f and g are functions evaluated at the data points $\mathbf{x}_i$ and w_i is the weight corresponding to $\mathbf{x}_i$, with $w_i > 0$.

The first step in generating the orthogonal polynomials is to find a set of monomial powers $\boldsymbol{\alpha}_j$ which spans the polynomial space on $\mathbf{x}_i$. This is done by starting with the monomial 1 and systematically adding monomials of increasing degree $\boldsymbol{\alpha}_j$. As each monomial is added, a matrix

$$X = \begin{bmatrix} \mathbf{x}_1^{\boldsymbol{\alpha}_1} & \mathbf{x}_2^{\boldsymbol{\alpha}_1} & \ldots & \mathbf{x}_N^{\boldsymbol{\alpha}_1} \\ \mathbf{x}_1^{\boldsymbol{\alpha}_2} & \mathbf{x}_2^{\boldsymbol{\alpha}_2} & \ldots & \mathbf{x}_N^{\boldsymbol{\alpha}_2} \\ \vdots & \vdots & & \vdots \\ \mathbf{x}_1^{\boldsymbol{\alpha}_n} & \mathbf{x}_2^{\boldsymbol{\alpha}_n} & \ldots & \mathbf{x}_N^{\boldsymbol{\alpha}_n} \end{bmatrix}$$

is generated for some initial value n. New rows are added to X for successive values of $\boldsymbol{\alpha}_j$, taken in lexicographical order at each $|\boldsymbol{\alpha}|$. The rank of X is checked at each step; if it is rank-deficient, the newly added monomial is rejected. Otherwise, it is added to the list of $\boldsymbol{\alpha}_j$ to be included in generating the polynomials. Rejection of a monomial power will happen because the point configuration is singular for the combination of monomials which would result from including the new $\boldsymbol{\alpha}_j$. Monomials are added until X is square and of full rank. The output of this procedure is a list of monomial powers which together span the polynomial space on the data points.

To generate the orthogonal polynomials from the list of monomials, the following procedure is used:

1. Generate the symmetric, positive definite matrix M, with $M_{ij} = \langle \mathbf{x}^{\boldsymbol{\alpha}_i}, \mathbf{x}^{\boldsymbol{\alpha}_j} \rangle$.
2. Perform the decomposition $M = SDS^T$, where D is a diagonal matrix and S is lower triangular.
3. Solve $S^T R = D^{-1/2}$, where $D^{-1/2} = \operatorname{diag}\{(d_1 w_1)^{-1/2}, \ldots, (d_N w_N)^{-1/2}\}$. This can be done using an LU solver with rearrangement of the matrix entries.
4. The matrix R now contains the coefficients of the orthogonal polynomials.

In implementing the method, we note that S^T can be found directly by using the algorithm given by Golub and Van Loan [5, page 138] with the row and column indices switched.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

The orthogonal polynomials P_i are now

$$P_i(\mathbf{x}) = \sum_{j=1}^{n} R_{ij}\mathbf{x}^{\boldsymbol{\alpha}_j},$$

and, for later convenience, we scale the coefficients on the inner products $\langle P_i(\mathbf{x})P_i(\mathbf{x})\rangle$ to give an orthonormal basis.

2.2. Fitting data on sets of scattered points. Given the set of orthogonal polynomials $P_i(\mathbf{x})$, the generation of a least-squares fit is trivial. By orthogonality,

$$f(\mathbf{x}) \approx \sum_i c_i P_i(\mathbf{x}), \tag{2.2}$$

where the constants c_i are given by

$$c_i = \sum_j f_j w_j P_i(\mathbf{x}_j).$$

Rearranging to give the interpolant as a weighted sum over the data points,

$$f(\mathbf{x}) \approx \sum_j v_j f_j, \tag{2.3}$$

$$v_j = w_j \sum_i P_i(\mathbf{x}_j)P_i(\mathbf{x}).$$

Derivatives of $f(\mathbf{x})$ can also be estimated as a weighted sum of the function values at the points of the distribution to generate a differentiation stencil:

$$\frac{\partial^{l+m+\cdots}}{\partial^l x_1 \partial^m x_2 \cdots} f(\mathbf{x}) \approx \sum_j v_j^{(lm\cdots)} f_j, \tag{2.4}$$

$$v_j^{(lm\cdots)} = w_j \sum_i P_i(\mathbf{x}_j) \frac{\partial^{l+m+\cdots}}{\partial^l x_1 \partial^m x_2 \cdots} P_i(\mathbf{x}),$$

with the derivatives of $P_i(\mathbf{x})$ being computed directly from the coefficients in the matrix R of section 2.1.

In summary, a derivative of a function $f(\mathbf{x})$ given on a set of points can be estimated at some point $\mathbf{x}_0$ using these steps:

1. Select N points in the region of $\mathbf{x}_0$, including $\mathbf{x}_0$ itself;
2. Generate a set of orthonormal polynomials for the selected points using the procedure of section 2.1;
3. Evaluate the weights $v_j^{(lm\cdots)}$ given by (2.4);
4. Calculate the derivative as the weighted sum of the function values.

An important point is that strictly, this procedure can only evaluate linear combinations of derivatives. In an extreme case, where only two points are used, the available monomials will be 1 and x_1 (or x_2 depending on the lexicographical ordering used). This allows linear interpolation of a function f between the two points and estimation of the derivative on a straight line joining them. This derivative will be a linear combination of $\partial f/\partial x_1$ and $\partial f/\partial x_2$, with the precise combination depending on the orientation of the two points. In practice, this should not be a serious limitation, since the monomials used in the polynomials are known and it is possible to determine whether there is a full set available for the determination of all derivatives of a given order.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

3. Performance. To illustrate the operation of the method, the first results presented are orthogonal polynomials on regular arrangements of points. The first is a regular 3×3 grid in $(-1,1) \times (-1,1)$. Upon application of the algorithm of section 2.1, the monomials which form the final matrix X are 1, x_1, x_2, x_1^2, x_1x_2, x_2^2, $x_1^2x_2$, $x_1x_2^2$, and $x_1^2x_2^2$, and the resulting orthogonal polynomials are

$$P_0 = \frac{1}{3}, \tag{3.1a}$$

$$P_1 = \frac{1}{2^{1/2}3^{1/2}}x_1, \tag{3.1b}$$

$$P_2 = \frac{1}{2^{1/2}3^{1/2}}x_2, \tag{3.1c}$$

$$P_3 = -\frac{2^{1/2}}{3} + \frac{1}{2^{1/2}}x_1^2, \tag{3.1d}$$

$$P_4 = \frac{1}{2}x_1x_2, \tag{3.1e}$$

$$P_5 = -\frac{2^{1/2}}{3} + \frac{1}{2^{1/2}}x_2^2, \tag{3.1f}$$

$$P_6 = -\frac{1}{3^{1/2}}x_2 + \frac{3^{1/2}}{2}x_1^2x_2, \tag{3.1g}$$

$$P_7 = -\frac{1}{3^{1/2}}x_1 + \frac{3^{1/2}}{2}x_1x_2^2, \tag{3.1h}$$

$$P_8 = \frac{2}{3} - x_1^2 - x_2^2 + \frac{3}{2}x_1^2x_2. \tag{3.1i}$$

If the procedure is applied to six points equally spaced on a unit circle, the resulting polynomials are

$$P_0 = \frac{1}{2^{1/2}3^{1/2}}, \tag{3.2a}$$

$$P_1 = \frac{1}{3^{1/2}}x_1, \tag{3.2b}$$

$$P_2 = \frac{1}{3^{1/2}}x_2, \tag{3.2c}$$

$$P_3 = -\frac{1}{3^{1/2}} + \frac{2}{3^{1/2}}x_1^2, \tag{3.2d}$$

$$P_4 = \frac{2}{3^{1/2}}x_1x_2, \tag{3.2e}$$

$$P_5 = -\frac{3^{1/2}}{2^{1/2}}x_1 + 2\frac{2^{1/2}}{3^{1/2}}x_1^3. \tag{3.2f}$$

A number of general issues are illustrated by these examples. The first is the obvious one that there are no more polynomials than there are points. This means that although the polynomials are notionally up to third order in both cases, in practice neither group of functions has a complete set of monomials capable of spanning all polynomials up to cubic. Secondly, if the orthogonal polynomials can be generated, there is no benefit in adding extra points once a complete set of functions is available: the result of adding more points is to yield an incomplete set of higher order polynomials. In applications, it may well be better to have a lower order, but complete, system to fit the functions on the points.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

3.1. Random point distributions on the unit disc or ball. The first set of results presented are average data for tests conducted with varying order and length scale. Following the example of Belward, Turner, and Ilić [1], the accuracy and robustness of the computational scheme are tested by estimating the derivatives of a prescribed function using a set of points randomly distributed over the unit disc or ball. The functions used are

$$f_1(\mathbf{x}) = R^4, \tag{3.3a}$$

$$f_2(\mathbf{x}) = \mathrm{e}^{-R^2}, \tag{3.3b}$$

$$f_3(\mathbf{x}) = x_1 \mathrm{e}^{-R^2}, \tag{3.3c}$$

$$R^2 = \sum_j x_j^2.$$

The functions have been chosen to give a function which can be fitted exactly by a polynomial (f_1), a Gaussian of the type found in various applications such as vortex dynamics (f_2), and a Gaussian weighted to give an asymmetry with a consequent nonzero first derivative at the evaluation point (f_3). The evaluation point was fixed at $\mathbf{0}$, and random points were distributed uniformly over the unit disc or ball.

In three dimensions, points were selected by randomly generating points (α, β, γ), with α, β, and γ uniformly distributed in the range $[-1, 1]$, and rejecting points lying outside the unit sphere. In two dimensions, a different approach was adopted in order to study the effect of point configuration on the performance of the method. In the test results presented in this section, points $(\alpha^{1/2} \cos 2\pi\gamma, \alpha^{1/2} \sin 2\pi\gamma)$ were taken, with both variables α and γ uniformly distributed in $[0, 1]$. Using $\alpha^{1/2}$ distributes points uniformly over the area of the disc. In section 3.2, it will be seen that using α^1 as a radius leads to numerical problems, as points are concentrated near the center of the disc.

Tests were conducted on 32 randomly generated point distributions in two and three dimensions, using unit weights $w_i = 1$ and varying the number of points used, the maximum order of polynomial, and the function scale. The number of points used was $N = 8, 16, 32, 64$, and 128, and the maximum order was set to 2, 3, and 4. The function scale σ was varied in order to examine the convergence of the derivative estimate. Without changing the coordinates of the points, the test function was evaluated as $f(\sigma\mathbf{x})$, with $\sigma = 2^a$, $a = -4, \ldots, 2$. The convergence rate of the estimator is then the gradient of $\log|\epsilon|$ against $\log\sigma$, with ϵ the error in the estimate.

Finally, a rescaling procedure was applied, equivalent to the non-dimensionalization employed in many applications [2]. In principle, this should improve the conditioning of the X matrix used to choose monomials for the fit although, as will be seen in the next section, this is by no means guaranteed. In generating the polynomials, the coordinates are scaled on a reference length Δ equal to the distance to the furthest point used in the polynomial fit. After evaluation, the derivatives are rescaled to give values in the original coordinates. The result returned, in other words, is

$$\frac{\partial^{l+m+\cdots}}{\partial^l x_1 \partial^m x_2 \cdots} f(\mathbf{x}) = \frac{1}{\Delta^{l+m+\cdots}} \frac{\partial^{l+m+\cdots}}{\partial^l x_1' \partial^m x_2' \cdots} f(\mathbf{x}'),$$
$$\mathbf{x}' = \mathbf{x}/\Delta.$$

Two sets of results are shown graphically, and the performance of all tests is given in tabular form.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

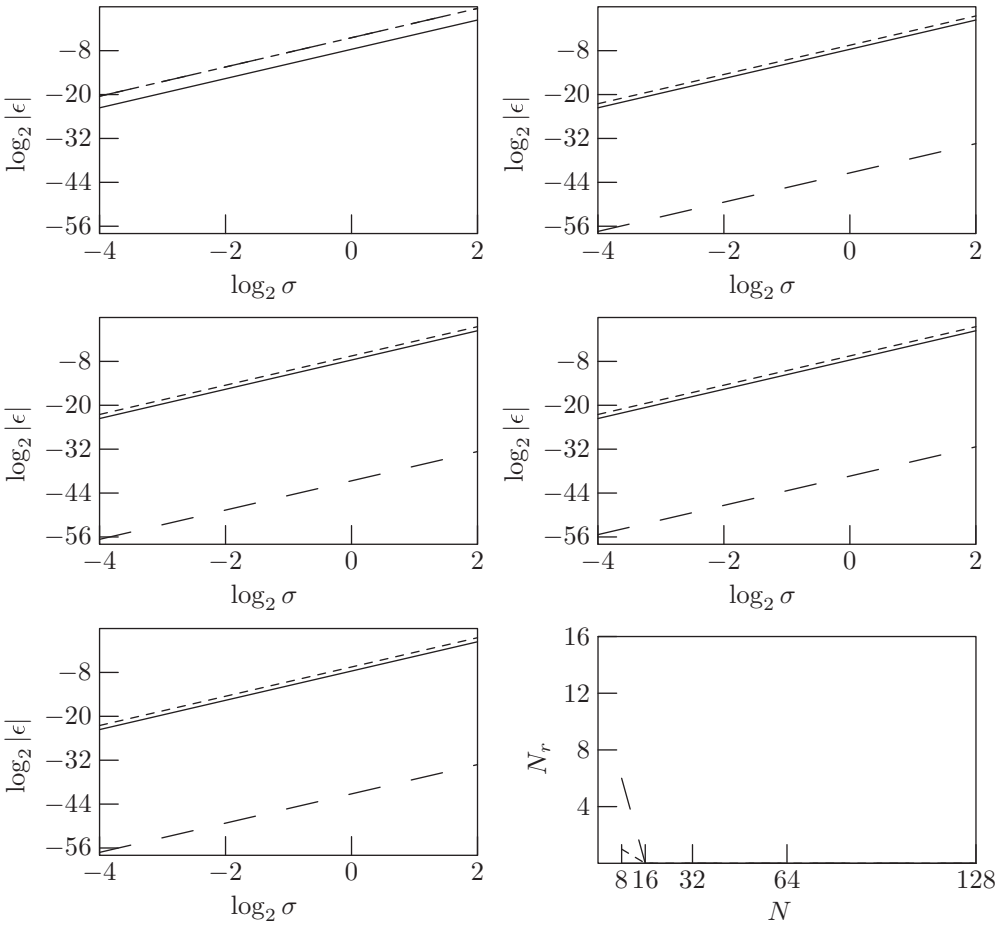

FIG. 3.1. *Evaluation of $\partial f_1/\partial x_1$ in two dimensions: mean absolute error against scale factor σ for, reading left to right, $N = 8, 16, 32, 64, 128$ points; final plot mean number of rejected monomials N_r against number of points N. Second order fit shown solid, third order dashed, and fourth order long dashed.*

Figure 3.1 shows the error in estimating $\partial f_1/\partial x_1$ in two dimensions, as a function of N and of polynomial order. The final plot shows N_r, the number of rejected monomials, as a function of N. This is independent of σ and depends only on the point configuration, to which it can be quite sensitive, as shown in section 3.2.

Table 3.1 gives the convergence rates and sample errors for evaluation of $\partial f_i/\partial x_1$ in the two-dimensional case. Convergence rates are given as a function of N and maximum polynomial order for each test function. The last two columns give the minimum and maximum error for $N = 128$ at each order.

Convergence for f_1 is fourth order in all cases, as might be expected from Figure 3.1. In the nonpolynomial cases, at small point number, the different order fits have the same convergence rate, since they are all essentially second order (with the higher order terms rejected). As N is increased for $\partial f_2/\partial x_1$, the fourth order fit becomes truly fourth order with a convergence rate slightly less than six for $\partial f_2/\partial x_1$, while the third order fit has the same convergence rate as the second order, since the third derivatives are zero at $\mathbf{0}$. For f_3, the situation is reversed, with the third and

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

TABLE 3.1
Convergence rates for first derivatives in two-dimensional problems.

		N					$\epsilon_{\min}^{(128)}$	$\epsilon_{\max}^{(128)}$
		8	16	32	64	128		
f_2	2	4.00	4.00	4.00	4.00	4.00	7.72×10^{-8}	1.29×10^{0}
	3	4.00	4.00	4.00	4.00	4.00	1.67×10^{-7}	2.80×10^{0}
	4	4.00	4.00	4.00	4.00	4.00	6.02×10^{-18}	1.01×10^{-10}
f_2	2	3.92	3.92	3.92	3.92	3.92	7.16×10^{-8}	8.00×10^{-1}
	3	3.92	3.92	3.92	3.92	3.92	5.37×10^{-7}	5.72×10^{0}
	4	3.92	5.84	5.84	5.84	5.84	4.94×10^{-11}	1.46×10^{0}
f_3	2	2.93	2.93	2.93	2.93	2.93	6.96×10^{-6}	1.27×10^{0}
	3	4.88	4.84	4.84	4.84	4.84	2.78×10^{-9}	1.26×10^{0}
	4	4.88	4.84	4.83	4.84	4.84	2.44×10^{-9}	1.07×10^{0}

TABLE 3.2
Convergence rates for second derivatives in two-dimensional problems.

	Order	N					$\epsilon_{\min}^{(128)}$	$\epsilon_{\max}^{(128)}$
		8	16	32	64	128		
f_1	2	4.00	4.00	4.00	4.00	4.00	5.04×10^{-6}	8.46×10^{1}
	3	4.00	4.00	4.00	4.00	4.00	4.41×10^{-6}	7.40×10^{1}
	4	4.00	4.00	4.00	4.00	4.00	2.16×10^{-16}	3.63×10^{-9}
f_2	2	3.95	3.95	3.95	3.95	3.95	9.20×10^{-7}	1.17×10^{1}
	3	3.90	3.92	3.92	3.92	3.92	2.91×10^{-6}	3.10×10^{1}
	4	3.90	5.88	5.88	5.88	5.88	2.20×10^{-10}	8.04×10^{0}
f_3	2	2.92	2.92	2.92	2.92	2.92	2.38×10^{-4}	4.02×10^{1}
	3	4.90	4.91	4.91	4.91	4.91	1.71×10^{-8}	1.13×10^{1}
	4	4.90	4.87	4.87	4.87	4.87	4.38×10^{-8}	2.27×10^{1}

fourth order fits converging at the same rate due to the absence of even derivatives. In this case, the convergence is slightly less than fifth order.

Table 3.2 shows similar results for the evaluation of $\partial^2 f_i/\partial x_1^2$. Again, the results for f_1 show the same convergence rate for all polynomial orders, while the convergence of the other two functions is affected by the nature of the test function. The convergence for the fourth order polynomial is again about sixth order for $\partial^2 f_2/\partial x_1^2$ and about fifth order for $\partial^2 f_3/\partial x_1^2$. The third order fit behaves like the second order fit for $\partial^2 f_2/\partial x_1^2$, due to the absence of the third derivative, and performs similarly to the fourth order fit for $\partial^2 f_3/\partial x_1^2$.

Figure 3.2 and Tables 3.3 and 3.4 show the equivalent results for three-dimensional tests, with $\partial^2 f_2/\partial x_1^2$ being used in the graphical illustration of performance. As before, the first five plots in Figure 3.2 show the error in the estimated derivative, while the final plot shows the number of rejected monomials as a function of N.

For smaller values of N where the higher order fits have the same number of available monomials as the second order, the error curves are similar. For $N \geq 64$, where all three fits can be fully specified, the error curve for the fourth order fit has a steeper slope at small σ than the other two. Note that the error behavior for the second and third order fits is similar due to the absence of third derivatives in f_2.

Tables 3.3 and 3.4 show the convergence rates and errors for the three-dimensional tests. The polynomial f_1 gives good convergence at a constant fourth order, as in the two-dimensional case. The other functions have variable orders of convergence, depending on N and the resulting number of available monomials. The fourth order fit converges, on average, at about fifth order for the derivatives of f_2, although

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

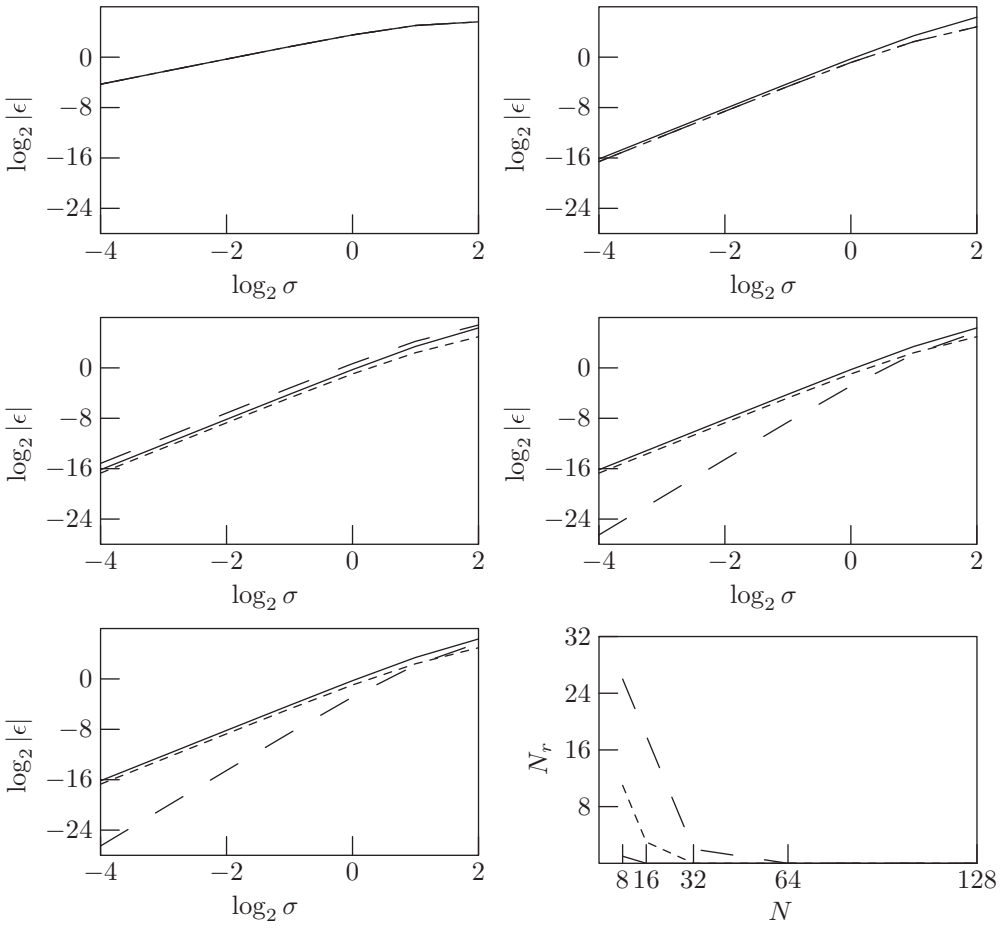

FIG. 3.2. *Evaluation of $\partial^2 f_2/\partial x_1^2$ in three dimensions: mean absolute error $|\bar{\epsilon}|$ against scale factor σ for, reading left to right, $N = 8, 16, 32, 64, 128$ points; final plot mean number of rejected monomials N_r against number of points N. Second order fit shown solid, third order dashed, and fourth order long dashed.*

TABLE 3.3
Convergence rates for first derivatives in three-dimensional problems.

		N					$\epsilon_{\min}^{(128)}$	$\epsilon_{\max}^{(128)}$
		8	16	32	64	128		
f_1	2	4.00	4.00	4.00	4.00	4.00	5.03×10^{-6}	8.43×10^{1}
	3	4.00	4.00	4.00	4.00	4.00	7.70×10^{-6}	1.29×10^{2}
	4	4.00	4.00	4.00	4.00	4.00	1.03×10^{-15}	1.73×10^{-8}
f_2	2	1.90	3.74	3.74	3.74	3.74	1.04×10^{-6}	4.42×10^{0}
	3	1.90	3.41	3.62	3.62	3.62	1.47×10^{-6}	3.38×10^{0}
	4	1.90	3.41	3.50	5.37	5.37	2.24×10^{-9}	6.26×10^{0}
f_3	2	2.66	2.73	2.73	2.73	2.73	7.16×10^{-5}	4.74×10^{0}
	3	2.66	2.69	4.50	4.50	4.50	3.80×10^{-8}	3.07×10^{0}
	4	2.66	2.69	4.29	4.29	4.29	2.13×10^{-6}	5.44×10^{1}

Figure 3.2 shows its convergence accelerating at small σ. The average convergence rate for f_3 is about the same, roughly fourth order, for third and fourth order fits, for the reasons mentioned above.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

TABLE 3.4
Convergence rates for second derivatives in three-dimensional problems.

		N					$\epsilon_{\min}^{(128)}$	$\epsilon_{\max}^{(128)}$
		8	16	32	64	128		
f_1	2	4.00	4.00	4.00	4.00	4.00	1.16×10^{-5}	1.94×10^{2}
	3	4.00	4.00	4.00	4.00	4.00	2.80×10^{-5}	4.71×10^{2}
	4	4.00	4.00	4.00	4.00	4.00	2.47×10^{-13}	4.14×10^{-6}
f_2	2	1.72	3.81	3.81	3.81	3.81	1.36×10^{-5}	8.12×10^{1}
	3	1.72	3.64	3.68	3.68	3.68	9.35×10^{-6}	3.08×10^{1}
	4	1.72	3.64	3.73	5.49	5.49	1.04×10^{-8}	5.25×10^{1}
f_3	2	2.70	2.75	2.75	2.75	2.75	6.43×10^{-4}	4.41×10^{1}
	3	2.70	2.78	4.48	4.48	4.48	1.51×10^{-7}	1.05×10^{1}
	4	2.70	2.78	4.41	4.25	4.25	5.13×10^{-7}	1.13×10^{1}

3.2. Effect of point distribution. A point which has been noted in previous work [2, 7] is the desirability of performing interpolations in nondimensional coordinates based on some local length scale, in order to improve the conditioning of the moving least squares method. In this paper, the local length scale Δ was defined as the distance from the evaluation point to the furthest point included in the polynomial fit. Some further analysis of a single test case will show some limitations of the method, even with this rescaling.

Figure 3.3 shows detailed results for tests carried out on two very similar point distributions. The test distributions had $N = 6, 7, \ldots, 19, 20$ points, sorted by distance from the evaluation point $\mathbf{x}_0 = \mathbf{0}$. A fourth order fit was used to estimate the three second derivatives of the Gaussian f_2 for each value of N and with $\sigma = 2^{-3}$. The point distributions used were generated from the same set of random numbers but in slightly different ways. The first set of points, "uniformly distributed in area," was generated using the same method as for the main tests of the previous section; the second, "uniformly distributed in radius," was generated using the same random numbers but with $\mathbf{x} = (\alpha \cos 2\pi\gamma, \alpha \sin 2\pi\gamma)$. The two point distributions are shown at the top of each column in Figure 3.3. It can be seen that they are very similar but the second distribution has its points bunched up toward the center, especially those points at small radius. The second line of the figure shows the scaling factor applied to the point distribution as a function of the number of points included.

This scaling behaves as one might expect: at large N, where the more distant points are included, $\Delta \approx 1$ for both point distributions, while at small N, the second point distribution has smaller radii and requires a larger scaling factor. The slight surprise comes in the next line of Figure 3.3, which shows the number of rejected monomials as a function of N. At small N, N_r decreases steadily in both cases and does so continuously for the points uniformly distributed in area. The second point distribution, however, requires that a monomial be rejected at $N = 11$ and at $N = 16$. The reason for this rejected monomial lies in the numerical properties of the point distribution, or stencil, independently of its scale. As the maximum radius, Δ, increases, the nondimensional values of the coordinates of low-numbered points, those near the origin, shrink. Since high powers (up to four) of these coordinates are used in determining which monomials are to be included in the polynomial fit and higher powers again (up to eight in this case) appear in the inner products in the matrix M, this leads to inevitable ill-conditioning. The result of this ill-conditioning can be seen in the error plots at the bottom of Figure 3.3: the first point distribution has errors which are roughly constant from the second order up to the start of the fourth

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

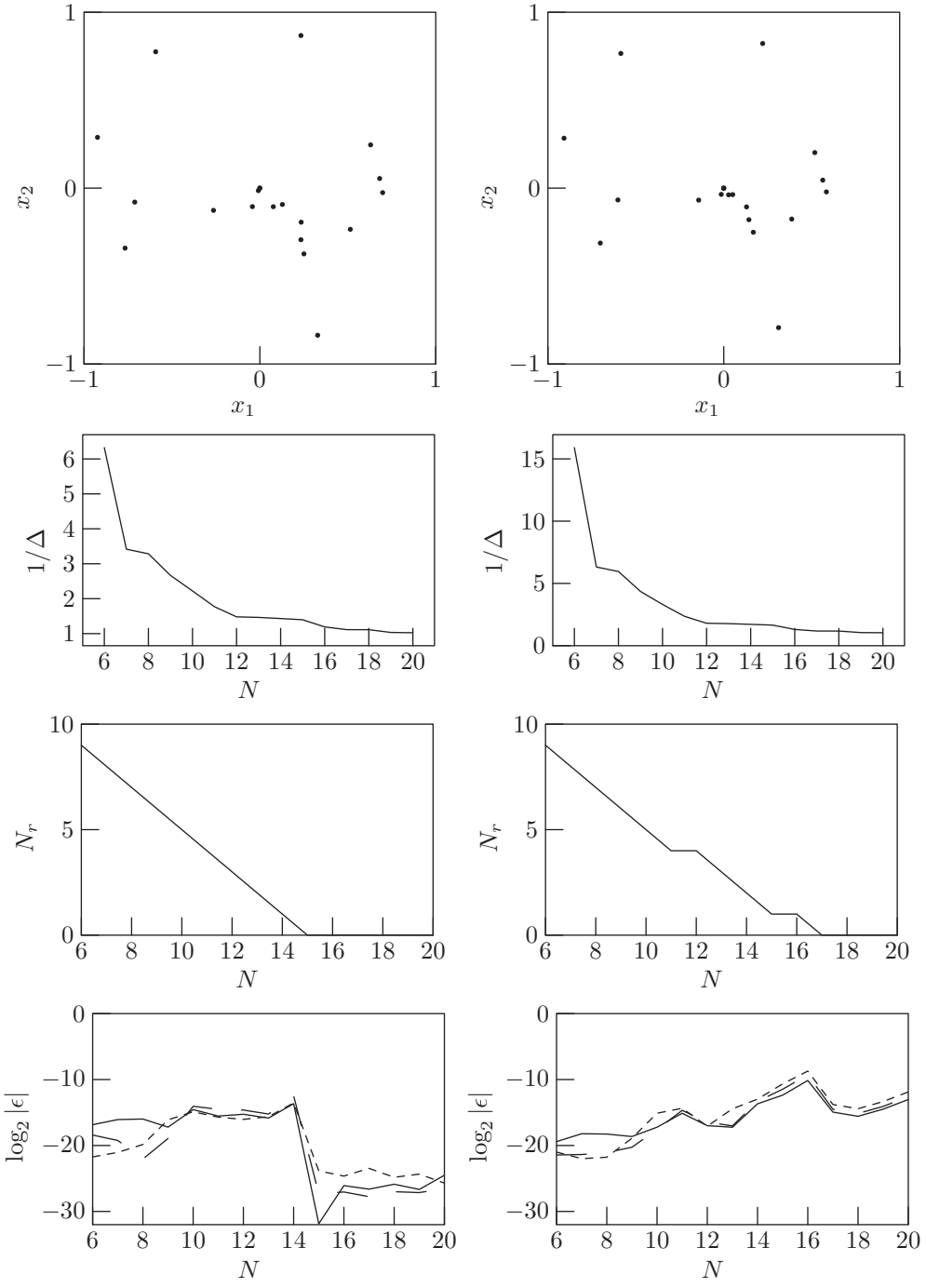

FIG. 3.3. *Single point configuration test: (top) point configurations; (second line) local length scale against number of points used; (third line) number of rejected monomials against number of points used; (bottom) error in* $\partial^2 f_2/\partial x_1^2$ *(solid),* $\partial^2 f_2/\partial x_1 \partial x_2$ *(dashed) and* $\partial^2 f_2/\partial x_2^2$ *(long dashed). The left-hand column of figures refers to the points uniformly distributed in area case while the right-hand column shows the same results for the points uniformly distributed in radius case.*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

TABLE 3.5
Coefficients for selected polynomials using different point distributions.

	P_5		P_9		P_{15}	
$x_1^0x_2^0$	1.88×10^{-1}	1.68×10^{-1}	1.05×10^{-1}	1.57×10^{-1}	7.35×10^{-1}	7.14×10^{-1}
$x_1^1x_2^0$	-6.34×10^{-1}	-1.17×10^{0}	-2.16×10^{0}	-1.07×10^{1}	7.50×10^{2}	2.56×10^{4}
$x_1^0x_2^1$	3.72×10^{0}	7.21×10^{0}	8.46×10^{0}	1.01×10^{1}	-3.22×10^{2}	-1.50×10^{4}
$x_1^2x_2^0$	1.14×10^{-1}	6.95×10^{-1}	1.25×10^{0}	1.38×10^{1}	-2.47×10^{3}	-1.91×10^{5}
$x_1^1x_2^1$	-4.21×10^{0}	-1.45×10^{1}	-2.06×10^{1}	-1.32×10^{2}	1.00×10^{4}	8.49×10^{5}
$x_1^0x_2^2$	6.02×10^{0}	1.88×10^{1}	5.23×10^{1}	8.77×10^{1}	3.34×10^{2}	-2.55×10^{5}
$x_1^3x_2^0$			1.00×10^{0}	2.81×10^{0}	1.16×10^{3}	3.19×10^{5}
$x_1^2x_2^1$			5.57×10^{0}	1.67×10^{2}	-2.04×10^{4}	-4.05×10^{6}
$x_1^1x_2^2$			-5.98×10^{1}	-1.64×10^{2}	3.14×10^{4}	6.88×10^{6}
$x_1^0x_2^3$			5.25×10^{1}	2.54×10^{2}	1.72×10^{4}	1.96×10^{6}
$x_1^4x_2^0$					6.21×10^{2}	-1.27×10^{5}
$x_1^3x_2^1$					1.12×10^{4}	4.33×10^{6}
$x_1^2x_2^2$					-3.73×10^{4}	-1.25×10^{7}
$x_1^1x_2^3$					-1.90×10^{4}	1.69×10^{6}
$x_1^0x_2^4$					3.37×10^{3}	1.85×10^{6}

order fit, dropping to a new, lower, value as the fourth order fit becomes available. In the case of the second point distribution, the ill-conditioning leads to an error which increases slightly with point number.

The effect of point distribution is made most clear by looking at the polynomials generated by the two different arrangements. Table 3.5 gives the coefficients for the first fully specified second, third, and fourth order polynomials on the "uniform in area" (left-hand columns) and "uniform in radius" (right-hand columns) distribution. The effect of changing the use of the random variables is clear: typically, the coefficients for the second distribution are an order of magnitude greater than their counterparts on the first. At higher order, the ill-conditioning introduced by the arrangement of points leads to polynomial coefficients up to three orders of magnitude greater than in the "uniform in area" case. The numerical difficulties can be avoided by increasing the tolerance of the rank test used to select monomials, but at the expense of rejecting more monomials. For a point configuration which causes difficulties, easily detected by tracking the number of monomials rejected, a low order fit on a small number of points near the origin gives better results than trying to use a higher order fit on a large number of points.

This also illustrates an advantage of the orthogonal polynomial approach over methods which employ singular value decomposition to generate basis functions. Chenoweth, Soria, and Ooi [2] discuss the problem of singular point configurations in the context of the singular value decomposition of a matrix containing the monomials evaluated at each point. As these authors note, a singular value decomposition shows which basis functions span the null space of polynomials on the points, allowing the detection of singular point configurations. The opposite fact is also true: the singular value decomposition yields a set of basis functions which span the function space on the points and, indeed, will indicate which of these basis functions are best determined. The problem, as we see above, is that even when a full set of well-determined

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

basis functions is available, it is not guaranteed that they form a suitable basis for the evaluation of derivatives, since they may lack the necessary monomials.

4. Conclusions. A method for moving least squares interpolation and differentiation using orthogonal polynomials has been presented and tested on random point distributions. The method makes use of the theory of discrete orthogonal polynomials in multiple variables and deals with the problems caused by singular point configurations by adjusting the terms of the polynomial. It is concluded that the method is robust and capable of detecting and compensating for singular configurations. In applications, it is recommended that the highest order of polynomial for which a full set of monomials is available be used in computing derivatives.

Acknowledgments. The author thanks the anonymous referees who read the original submission with great care, making many useful comments on the method of the paper and on its presentation.

REFERENCES

[1] J. A. Belward, I. W. Turner, and M. Ilić, *On derivative estimation and the solution of least squares problems*, J. Comput. Appl. Math., 222 (2008), pp. 511–523.
[2] S. K. M. Chenoweth, J. Soria, and A. Ooi, *A singularity-avoiding moving least squares scheme for two-dimensional unstructured meshes*, J. Comput. Phys., 228 (2009), pp. 5592–5619.
[3] G. E. Fasshauer, *Multivariate Meshfree Approximation*, Course notes, http://www.math.iit.edu/~fass/603.html.
[4] M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, M. Booth, and F. Rossi, *GNU Scientific Library Reference Manual*, Network Theory Ltd., Bristol, United Kingdom, 2005.
[5] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1996.
[6] D. Levin, *The approximation power of moving least-squares*, Math. Comp., 67 (1998), pp. 1517–1531.
[7] J. S. Marshall, J. R. Grant, A. A. Gossler, and S. A. Huyer, *Vorticity transport on a Lagrangian tetrahedral mesh*, J. Comput. Phys., 161 (2000), pp. 85–113.
[8] C. Moussa and M. J. Carley, *A Lagrangian vortex method for unbounded flows*, Internat. J. Numer. Methods Fluids, 52 (2008), pp. 161–181.
[9] W. Schönauer and T. Adolph, *How we solve PDEs*, J. Comput. Appl. Math., 131 (2001), pp. 473–492.
[10] W. Schönauer and T. Adolph, *Higher order may be better or may not be better: Investigations with the FDEM (finite difference element method)*, J. Sci. Comput., 17 (2002), pp. 221–229.
[11] Y. Xu, *Lecture notes on orthogonal polynomials of several variables*, in Inzell Lectures on Orthogonal Polynomials, Adv. Theory of Special Funct. Orthogonal Polynomials 2, W. zu Castell, F. Filbir, and B. Forster, eds., Nova Science, Hauppauge, NY, 2004, pp. 135–188.
[12] Y. Xu, *On discrete orthogonal polynomials of several variables*, Adv. Appl. Math., 33 (2004), pp. 615–632.
[13] Z. Zhang, P. Zhao, and K. M. Liew, *Analyzing three-dimensional potential problems with the improved element-free Galerkin method*, Comput. Mech., 44 (2009), pp. 273–284.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.